

# Introduction

*Project title:* **Sentiment Analysis API**

*Description:*

Building a backend service that exposes RESTful API endpoint for sentiment analysis. The API will accept text input and return the analyzed result using a pre-trained machine learning model.

Implemented a web server using Python web framework Flask. Created a single endpoint */analyze* that accepts POST requests. Text of user input “Text to be analyzed” for sentiment analysis performed by using a pre-trained machine learning model from Hugging Face Transformers Library.

Then the sentiment analysis returns a JSON response a results with structure

```
{  
    'sentiment': "Positive/Negative/Neutral"  
}
```

# Table of Contents

<b>Introduction.....</b>	<b>1</b>
Table of Contents.....	2
Project Structure.....	3
Run The Project.....	4
The Outputs (Design, Code).....	7
Evaluation.....	18

[Introduction](#)

[Table of Contents](#)

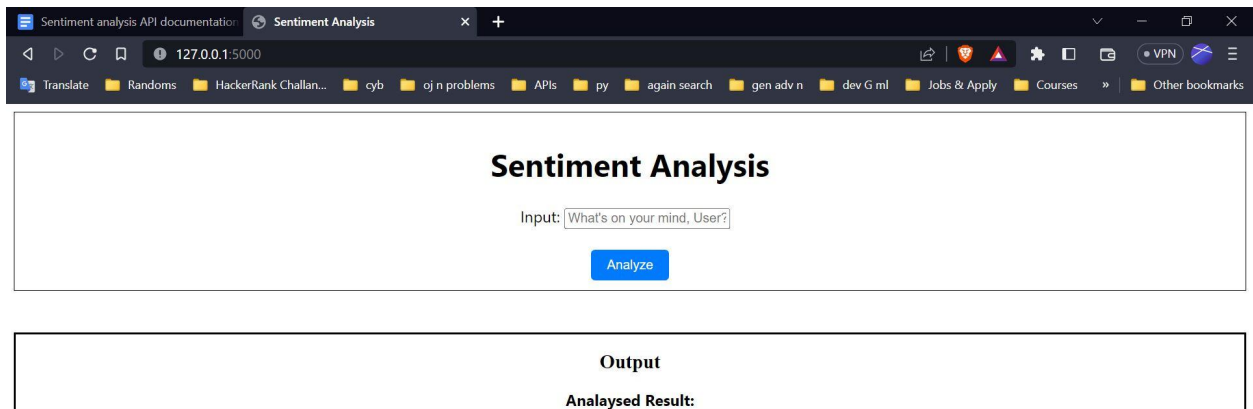
[Project Structure](#)

[Run The Project](#)

[The Outputs \(Design, Code\)](#)

[Evaluation](#)

## Web Application Interface Preview









Following section will contain **project structure**, **how to run the project locally** and **the output** section.

## Project Structure




Make a folder and name it 'sentiment\_analysis\_api' at your preferred directory on the computer.





Then make the subfolders within it.

Name	Date modified	Type	Size
 __pycache__	6/23/2023 3:07 AM	File folder	
 static	6/22/2023 4:26 PM	File folder	
 templates	6/22/2023 4:40 PM	File folder	
 venv	6/22/2023 4:28 PM	File folder	
 app.py	6/23/2023 4:01 AM	Python Source File	2 KB
 read_me.txt	6/23/2023 3:50 AM	Text Document	2 KB

The subfolder 'static' has a subfolder named 'css' and it contains a file named 'styles.css' which is for styling the input and output of html template.

 static	6/22/2023 4:26 PM	File folder	
 css	6/22/2023 4:26 PM	File folder	
 styles.css	6/23/2023 4:01 AM	CSS Source File	1 KB

The subfolder 'templates' has only file 'home.html'

 templates	6/22/2023 4:40 PM	File folder	
 home.html	6/23/2023 4:02 AM	Chrome HTML Docu...	2 KB

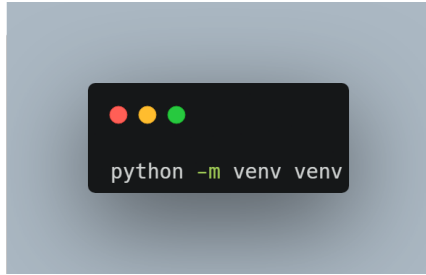
Then there is a 'app.py' file containing the convenient python codes to meet the requirements. And a 'read\_me.txt' file with necessary details for running this application locally.

*Following contains 'How to run the project locally'.*

## Run The Project

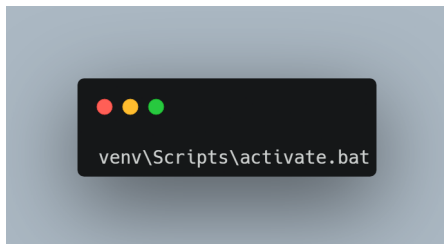
### # Step 1

Create a virtual environment within the project folder.



## # Step 2

Activate the virtual environment (Windows)



Mac/Linux



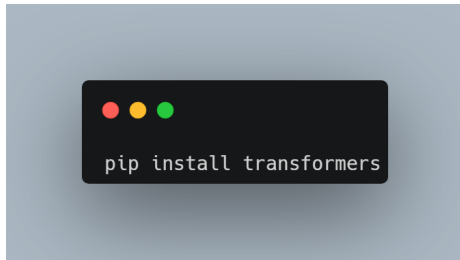
## # Step 3

Install flask in the virtual environment



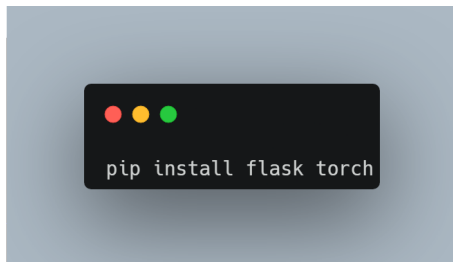
## # Step 4

Install transformers in virtual environment



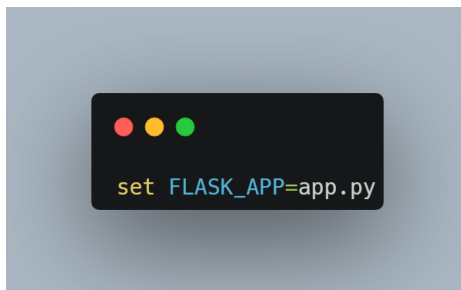
## # Step 5

Install torch to perform the transformers and pipeline



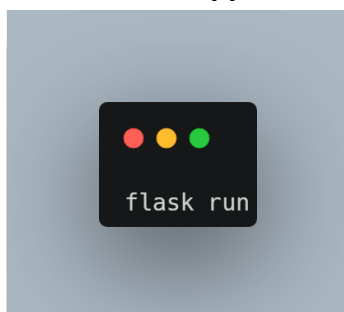
## # Step 6

Set the flask app in the virtual environment



## # Step 7

Run the flask app



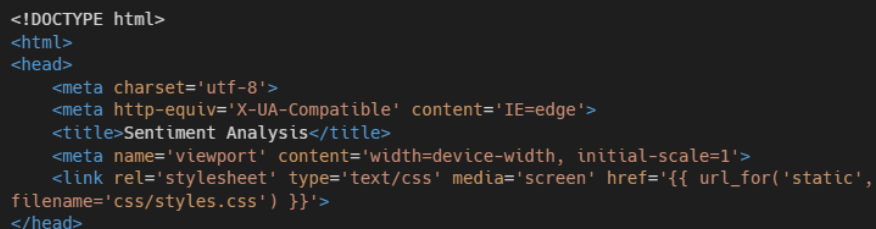
Then one can open the browser on computer and write in search bar <http://127.0.0.1:500>  
It will take user to the sentiment analysis web app's interface.

## The Outputs (Design, Code)

### HTML Template (home.html)

#### This head section of home.html

contains information that helps a web browser display a webpage correctly. Includes instructions for specifying the character encoding of the webpage, specifying the title of the webpage, and providing a stylesheet to format the content of the page. Also specifies how the webpage should be displayed on different devices with varying screen sizes.



```
<!DOCTYPE html>
<html>
<head>
  <meta charset='utf-8'>
  <meta http-equiv='X-UA-Compatible' content='IE=edge'>
  <title>Sentiment Analysis</title>
  <meta name='viewport' content='width=device-width, initial-scale=1'>
  <link rel='stylesheet' type='text/css' media='screen' href='{{ url_for('static',
filename='css/styles.css') }}'>
</head>
```

#### The body part of the home.html

create a webpage with a form for sentiment analysis. The webpage has two main sections: the input section and the output section.

The input section contains a text box and a button for submitting text to be analyzed. If there was an error during submission, an error message will be displayed on the page.

The output section displays the analyzed results. The sentiment result will be displayed as text on the page. If available, the sentiment result will also be shown in JSON format below the text.

```
1 <body>
2
3 <!-- Text Input -->
4 <div class="input_div">
5
6 <!-- Error Message -->
7 {% if error_msg %}
8 <p> {{ error_msg }}</p>
9 {% endif %}
10
11 <h1>Sentiment Analysis</h1>
12 <form action="/analyze" method="POST">
13 <label for="text">Input: </label>
14 <input type="text" name="text" placeholder="What's on your mind, User?" autocomplete="off"
required>
15 <br><br>
16 <button type="submit">Analyze</button>
17 </form>
18 </div>
19
20 <!-- Result Output -->
21 <br><br>
22 <div class="result_div">
23 <h2>Output</h2>
24 <b>Analysed Result: </b>
25 {% if sentiment %}
26 <p> {{ sentiment }} </p>
27 <div class="sentiment_jsonview">
28 <br>
29 <p>Input Text: </p> <b>{'text': '{{ json_text['text'] }}' }</b>
30 <p>Sentiment in JSON</p>
31 <b> <p>{ 'sentiment': ' {{ sentiment_json['sentiment'] }} ' }</p> </b>
32 </div>
33 {% endif %}
34 </div>
35
36 </body>
37
38 </html>
```

## CSS Styling (styles.css)

This is a Cascading Style Sheet (CSS) file used to define the visual styles for the web page associated with it.

- 1) Set the font family and text alignment for the entire webpage.




- 2) define the styles for different sections of the page, such as the input section and the result output section. For example, the input Div has a border, padding, and margins set up to create an outline around the input field. Similarly, the result Div has its own border and font size defined.
- 3) Includes specific instructions for styling the submit button. It has specified the background color, font color, padding, border radius, and cursor type.
- 4) Change the background color when the button is hovered over with the mouse.
- 5) Display the sentiment result in JSON format.

These rules specify the border, font size, and font family of the text displayed.

Overall, this CSS file provides instructions on how to display various elements of the webpage and ensure that they look consistent and visually appealing to the user.



```
body {  
  text-align: center;  
  font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;  
}  
  
/* Input */  
.input_div {  
  border: 1px solid black;  
  padding: 10px;  
}  
  
.input_div form input {  
  margin-right: 10px;  
}  
  
/* Submit Button (Analyze) */  
button {  
  
  background-color: #007bff;  
  color: white;  
  border: none;  
  padding: 8px 16px;  
  border-radius: 4px;  
  cursor: pointer;  
}
```



```
button:hover{
    background-color: green;
}

/* Result */
.result_div {
    border: 2px solid black;
}

.result_div h2{
    font-size: 20px;
    font-family: 'Times New Roman', Times,
    Serif;
}


.result_div p {
    font-size: 16px;
    font-family: 'Times New Roman', Times,
    Serif;
}

/* Result in JSON Form */
.sentiment_jsonview {
    border: 2px solid black;
}

.sentiment_jsonview b {
    font-size: 16px;
}
```


## Sentiment Analysis Backend Code (app.py)

### # Importing the necessary modules and libraries



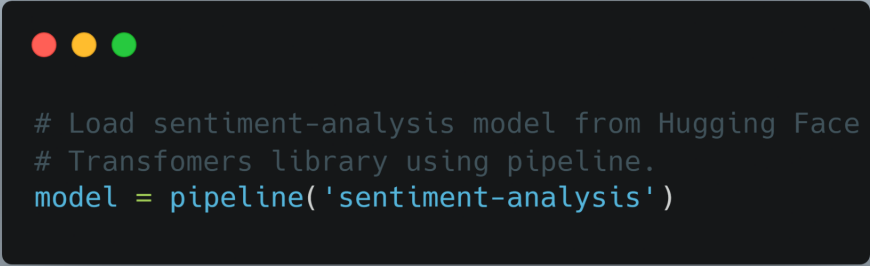
```
# Import flask modules and transformers pipeline.  
from flask import Flask, jsonify, json, render_template, request  
from transformers import pipeline
```

### # Make Instances of Flask App



```
# Create an instance of Flask app.  
app = Flask(__name__)
```

### # Model of Hugging Face Transformer Pipeline



```
# Load sentiment-analysis model from Hugging Face
# Transformers library using pipeline.
model = pipeline('sentiment-analysis')
```

#### # Route of home page



```
# Route of home page
@app.route('/')
def home():
    return render_template('home.html')
```

#### # Route of sentiment analysis api (/analyze)

```
1
2 # Route of sentiment analysis api
3 @app.route('/analyze', methods=['POST'])
4 def sentiment_analysis():
5     try:
6         # The text to be analyzed from the form data
7         text = request.form['text']
8
9         # Analyze the sentiment of text using pre-trained model
10        result = model(text)[0]
11        sentiment = result['label']
12
13        sentiment_jsonify = {}
14        # text_jsonify = {}
15
16        # Create JSON response.
17        json_text = {'text': text}
18        json_data = {'sentiment': sentiment}
19
20        # Print JSON response to command prompt
21        sentiment_jsonify = json_data
22        print(json.dumps(json_data))
23
24        # Return the rendered template with sentiment and text
25        return render_template('home.html', sentiment=sentiment, sentiment_json=sentiment_jsonify,
26                               json_text=json_text)
27    except Exception as e:
28        # If an exception occurs, display an error message in the template
29        error_msg = str(e)
30        return render_template('home.html', error_msg=error_msg)
31
```

## # Run the Flask app

```
1 # Run the Flask app
2 if __name__ == '__main__':
3     app.run(debug=True)
4
```

## Output after running the sentiment analysis.

Sample Text Input	Sample Sentiment Output
Thank you for the opportunity.	Positive
I disliked the theme.	Positive
I liked the movie but the seats were uncomfortable.	Negative
I like playing football and cricket.	Positive
There is a road by our house.	Positive
I love drinking cold water in the hot summer.	Positive
I enjoyed the food but rude behavior of the staff and disgusting surroundings I did not like.	Negative
I have a good laptop.	Positive
The sky is blue today.	Positive

**Some of analyzed result demo on the web application:**

**Test text1:** Thank you for the opportunity.

**Sentiment result: Positive**

*[Pictorial Evidence]*

Sentiment analysis API documentationSentiment Analysis

127.0.0.1:5000/analyze

TranslateRandomsHackerRank Challan...cyboj n problemsAPIspyagain searchgen adv ndev G mlJobs & ApplyCoursesOther bookmarks

## Sentiment Analysis

Input:

Analyze

### Output

**Analysed Result:**

POSITIVE

Input Text:

`{'text': 'Thank you for the opportunity.'}`

Sentiment in JSON

`{ 'sentiment': ' POSITIVE ' }`

**Test text2:** The sky is blue today.

**Sentiment result: Positive**

*[Pictorial Evidence]*

Sentiment analysis API documentationSentiment Analysis

127.0.0.1:5000/analyze

TranslateRandomsHackerRank Challan...cyboj n problemsAPIspyagain searchgen adv ndev G mlJobs & ApplyCoursesOther bookmarks

## Sentiment Analysis

Input:

Analyze

### Output

**Analysed Result:**

POSITIVE

Input Text:

`{'text': 'The sky is blue today.'}`

Sentiment in JSON

`{ 'sentiment': ' POSITIVE ' }`

**Test text3:** I enjoyed the food but rude behavior of the staff and disgusting surroundings I did not like.



## Sentiment result: Negative

[Pictorial Evidence]

**Sentiment Analysis**

Input:

**Analyze**

**Output**

**Analysed Result:**

NEGATIVE

Input Text:

{ 'text': 'I enjoyed the food but rude behavior of the staff and disgusting surroundings I did not like.' }

Sentiment in JSON

{ 'sentiment': 'NEGATIVE ' }

Test text4: I disliked the theme.

## Sentiment result: Negative

[Pictorial Evidence]

**Sentiment Analysis**

Input:

**Analyze**

**Output**

**Analysed Result:**

NEGATIVE

Input Text:

{ 'text': 'I disliked the theme.' }

Sentiment in JSON

{ 'sentiment': 'NEGATIVE ' }

# Evaluation

To evaluate a program it is important to consider both its accuracy and limitations.

Sentiment analysis program using Hugging Face Transformers library is able to correctly identify positive and negative sentiment; this struggle with neutral statements may indicate that it is not effective at distinguishing between ambiguous or mixed emotion.

To Improve the performance of sentiment analysis program:

- 1) More sophisticated ML algorithms can be used. While the Hugging Face Transformers library is a powerful resource for NLP, it may be good to explore other ML techniques to improve accuracy like CNNs or LSTM models.
- 2) Fine tuning sentiment analysis program model by modifying hyperparameters or tweaking its architecture. Adjusting number of layers, changing activation function or different learning rates to improve its ability to identify neutral sentiment.

Thank you.

Md. Rejoan Siddiky

siddikyrejoan@gmail.com

